Policy iteration in finite templates domain

Assalé Adjé*
DTIM, Onera
31055 Toulouse Cedex 4 - France.
email:assale.adje@onera.fr

Abstract

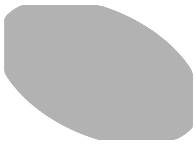
We prove in this paper that policy iteration can be generally defined in finite domain of templates using Lagrange duality. Such policy iteration algorithm converges to a fixed point when for very simple technique condition holds. This fixed point furnishes a safe over-approximation of the set of reachable values taken by the variables of a program. We apply our method in the case of quadratic templates to compute ellipsoid numerical invariants using semi-definite programming.

1 Introduction

We introduce a complete lattice consisting of sub-level sets of (possibly non-convex) functions, which we use as an abstract domain in the sense of abstract interpretation [CC77] for computing numerical program invariants. This abstract domain is parametrized by a basis of functions, akin to the approach put forward by Manna, Sankaranarayanan, and Sipma (the linear template abstract domain [SSM05] see also [SCSM06]), except that the basis functions or templates which we use here need not be linear. The domains obtained in this way encompass the classical abstract domains of intervals, octagons and linear templates.

To illustrate the interest of this generalization, let us consider a harmonic oscillator: $\ddot{x} + c\dot{x} + x = 0$. By taking an explicit Euler scheme, and for $c = 1$ we get the program shown at the left of Figure 1.

```
x = [0,1];
v: = [0,1];
h = 0.01;
while (true) { [2]
  w = v;
  v = v*(1-h)-h*x;
  x = x+h*w; [3] }
```



$\{x^2 \leq 3.5000,\ v^2 \leq 2.3333,\ 2x^2 + 3v^2 + 2xv \leq 7\}$

Figure 1: Euler integration scheme of a harmonic oscillator and the loop invariant found at control point 2

The invariant found with our method is shown right of Figure 1. For this, we have considered the template based on functions $\{x^2, v^2, 2x^2+3v^2+2xv\}$, i.e. we consider a domain where we are looking for upper bounds of these quantities. This means that we consider the *non-linear* quadratic homogeneous templates based on $\{x^2, v^2\}$, i.e. symmetric intervals for each variable of the program, together with the *non-linear* template $2x^2 + 3v^2 + 2xv$. The last template comes from a Lyapunov function that the designer of the *algorithm* may have considered to prove the stability of his scheme, *before it has been implemented*. This allows us to represent set defined by constraints of the form $x^2 \leq c_1$, $v^2 \leq c_2$ and $2x^2 + 3v^2 + 2xv \leq c_3$ where c_1, c_2 and c_3 are degrees of freedom. In view of *proving the implementation correct*, one is naturally led to considering such templates. Last but not least, it is to be noted that the loop invariant using intervals, zones, octagons or even polyhedra (hence with any linear template) is the insufficiently strong invariant $h = 0.01$ (the variables v and x cannot be bounded.) However, the main interest of the present method is to carry over to the non-linear setting.

*This work is supported by the RTRA / STAE Project BRIEFCASE.

Contributions of the paper We (recall) describe the lattice theoretical operations in terms of Galois connections and generalized convexity in Section 2. We next introduce an abstract semantics $F^\sharp$, and a *relaxed* semantics $F^{\mathcal{R}}$ using Lagrange duality. The relaxed semantics is, by construction, an over-approximation of the abstract one. We prove that the relaxed semantics has more powerful properties that explains why we should consider this latter. Moreover, we show in Subection 5.3 that the vectors of Lagrange multipliers produced by this relaxation correspond to policies, in a policy iteration technique for finding fixed points of $F^{\mathcal{R}}$, Theorem 5.3, precisely over-approximating the fixed points of $F^\sharp$.

In Section 7, we apply our techniques in the case of a basis of quadratic functions, $F^{\mathcal{R}}$ can be computed by solving a semi-definite program using Shor's relaxation. The advantage of the latter is that it can be solved in polynomial time to an arbitrary prescribed precision by the ellipsoid method [GLS88], or by interior point methods [NN94] if a strictly feasible solution is available (however, we warn the reader that interior points methods, which are more efficient in practice, are known to be polynomial only in the real number model, not in the bit model, see the survey [PR97] for more information). Finally, we illustrate the method on the running example.

Related work This work is to be considered as a generalisation of [SSM05], [SCSM06] because it extends the notion of templates to non-linear functions, and of [CGG$^+$05], [GGTZ07], [AGG08], [GS07a] and [GS07b] since it also generalizes the use of policy iteration for better and faster resolution of abstract semantic equations. Polynomial inequalities (of bounded degree) were used in [BRCZ05] in the abstract interpretation framework but the method relies on a reduction to linear inequalities (the polyhedra domain), hence is more abstract than our domain. Particular quadratic inequalities (involving two variables - i.e. ellipsoidal methods) were used for order 2 linear recursive filters invariant generation in [Fer05][1]. Polynomial *equalities* (and not general functional inequalities as we consider here) were considered in [MOS04, RCK07]. The use of optimization techniques and relaxations for program analysis has also been proposed in [Cou05], mostly for synthesising variants for proving termination, but invariant synthesis was also briefly sketched, with different methods than ours (concerning the abstract semantics and the fixpoint algorithm). Finally, the interest of using quadratic invariants and in particular Lyapunov functions for proving control programs correct (mostly in the context of Hoare-like program proofs) has also been advocated recently by E. Féron et al. in [FF08, FA08].

2 Recalling the generalized templates

In [AGG10, AGG11], we introduced the concept of generalized templates which are just functions from $\mathbb{R}^d$ to $\mathbb{R}$. In this paper, we are interesting in developing a generalized policy iteration to compute numerical invariants in a templates context.

2.1 Basic notions

Let X, Y two non empty sets. We denote by $\mathbf{F}(X, Y)$ the set of maps (functions when Y is a set of reals) from X to Y.

Let $\mathbb{P}$ be a subspace of $\mathbf{F}(\mathbb{R}^d, \mathbb{R})$. We are interested in replacing the classical concrete semantics by meaning of sublevel sets i.e. we have a functional representation of numerical invariants through the functions of $\mathbb{P}$. An invariant will be determined as the intersection of sublevel sets. The problem is thus reduced to find the interesting functions $p \in \mathbb{P}$ and the optimal levels on each templates p.

In consequence, we introduce a set of functions from $\mathbb{P}$ to $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty\} \cup \{+\infty\}$. We denote this set of functions by $\mathbf{F}(\mathbb{P}, \overline{\mathbb{R}})$. For an element $v \in \mathbf{F}(\mathbb{P}, \overline{\mathbb{R}})$, we associate the intersection of sublevel sets defined by $v(p)$ where p belongs to $\mathbb{P}$.

Definition 2.1 ($\mathbb{P}$-sublevel sets)**.** To a function $v \in \mathbf{F}(\mathbb{P}, \overline{\mathbb{R}})$, we associate the $\mathbb{P}$-sublevel set denoted by $v^\star$ and defined as:

$$\begin{aligned} v^\star &= \{x \in \mathbb{R}^d \mid p(x) \leq v(p),\ \forall p \in \mathbb{P}\} \\ &= \textstyle\bigcap_{p \in \mathbb{P}} \{x \in \mathbb{R}^d \mid p(x) \leq v(p)\} \end{aligned}$$

When $\mathbb{P}$ is a set of convex functions, the $\mathbb{P}$-sublevel sets corresponds to the intersection of classical sublevel sets from convex analysis. In our case, $\mathbb{P}$ can contain non-convex functions so $\mathbb{P}$-sublevel sets are not necessarily convex in the usual sense.

[1] A generalization to order n linear recursive filters is also sketched in this article.

We also want a functional representation of a set. In convex analysis, it is well-known that a closed convex set can be represented by its support function i.e. the supremum of linear forms on the set (e.g see Section 13 of [Roc96]). Here, we use the same notion but we replace the linear forms by the functions $p \in \mathbb{P}$ which are not necessarily linear. This generalization is not new and was introduced by Moreau [Mor70]. The reader can be also consult [Rub00, Sin97] for more details about those concepts.

Definition 2.2 ($\mathbb{P}$-support functions)**.** To $X \subseteq \mathbb{R}^d$, we associate the abstract support function denoted by $X^\dagger$ and defined as:

$$X^\dagger(p) = \sup_{x \in X} p(x)$$

The $\mathbb{P}$-support functions are always lower-semicontinuous (l.s.c.) i.e. all its sublevel sets are topologically closed and positively homogeneous i.e. the image commutes with nonnegative scalars. If $\mathbb{P}$ is a convex subset of convex functions then $\mathbb{P}$-support functions are convex on $\mathbb{P}$.

We equip the $\mathbf{F}\left(\mathbb{P}, \overline{\mathbb{R}}\right)$ with the classical partial order for the functions i.e $v \leq w \iff v(p) \leq w(p)$ for all $p \in \mathbb{P}$. We order the set of the subsets of $\mathbb{R}^d$ by the inclusion. By taking these orders, we get the following proposition.

Proposition 2.1. *The pair of maps $v \mapsto v^\star$ and $X \mapsto X^\dagger$ defines a Galois connection between $\mathbf{F}\left(\mathbb{P}, \overline{\mathbb{R}}\right)$ and the set of subsets of $\mathbb{R}^d$.*

In the terminology of abstract interpretation, $(.)^\dagger$ is the abstraction function, and $(.)^\star$ is the concretization function. The Galois connection result will provide the correctness of the semantics.

2.2 The lattices of $\mathbb{P}$-convex sets and $\mathbb{P}$-convex functions

Now, we are interested in closed elements (in term of Galois connection) that we call here $\mathbb{P}$-convex elements. Formally, they are defined as follows.

Definition 2.3 ($\mathbb{P}$-convexity)**.** Let $v \in \mathbf{F}\left(\mathbb{P}, \overline{\mathbb{R}}\right)$, we say that v is a $\mathbb{P}$-convex function if $v = (v^\star)^\dagger$. A set $X \subseteq \mathbb{R}^d$ is a $\mathbb{P}$-convex set if $X = (X^\dagger)^\star$.

Definition 2.4. We respectively denote by $\mathrm{Vex}_{\mathbb{P}}(\mathbb{P} \mapsto \overline{\mathbb{R}})$ and $\mathrm{Vex}_{\mathbb{P}}(\mathbb{R}^d)$ the set of $\mathbb{P}$-convex function of $\mathbf{F}\left(\mathbb{P}, \overline{\mathbb{R}}\right)$ and the set of $\mathbb{P}$-convex sets of $\mathbb{R}^d$.

The family of functions $\mathrm{Vex}_{\mathbb{P}}(\mathbb{P} \mapsto \overline{\mathbb{R}})$ is ordered by the partial order of real-valued functions i.e $v \leq w \iff v(p) \leq w(p)\ \forall p \in \mathbb{P}$. The family of set $\mathrm{Vex}_{\mathbb{P}}(\mathbb{R}^d)$ is ordered by the inclusion order denoted by $\subseteq$. Galois connection permits to construct lattice operations on $\mathbb{P}$-convex elements. They are defined as follows.

Definition 2.5 (The meet and join)**.** Let v and w be in $\mathbf{F}\left(\mathbb{P}, \overline{\mathbb{R}}\right)$. We denote by $\inf(v, w)$ and $\sup(v, w)$ the functions defined respectively by, $p \mapsto \inf(v(p), w(p))$ and $p \mapsto \sup(v(p), w(p))$. We equip $\mathrm{Vex}_{\mathbb{P}}(\mathbb{P} \mapsto \overline{\mathbb{R}})$ with the meet (respectively join) operator:

$$v \vee w = \sup(v, w) \tag{1}$$

$$v \wedge w = (\inf(v, w)^\star)^\dagger \tag{2}$$

Similarly, we equip $\mathrm{Vex}_{\mathbb{P}}(\mathbb{R}^d)$ with the two following operators: $X \sqcup Y = ((X \cup Y)^\dagger)^\star$, $X \sqcap Y = X \cap Y$.

It is well-known that with the prevois lattice operations, the lattice sets of $\mathbb{P}$-convex elements are isomorphic complete lattices.

Theorem 2.1. $(\mathrm{Vex}_{\mathbb{P}}(\mathbb{P} \mapsto \overline{\mathbb{R}}), \wedge, \vee)$ *and* $(\mathrm{Vex}_{\mathbb{P}}(\mathbb{R}^d), \sqcap, \sqcup)$ *are isomorphic complete lattices.*

We end this section by specifying the closure operator in our context.

Definition 2.6. For $v \in \mathbf{F}\left(\mathbb{P}, \overline{\mathbb{R}}\right)$, we denote by $\mathrm{vex}_{\mathbb{P}}(v)$ the $\mathbb{P}$-convex hull of v which is the greatest $\mathbb{P}$-convex function smaller than v.

Similarly, we denote by the set $\mathrm{vex}_{\mathbb{P}}(X)$ the $\mathbb{P}$-convex hull of a subset X which is the smallest $\mathbb{P}$-convex set greater than X.

3 Abstract and Relaxed semantics

Suppose now we are given a program with d variables $(x_1, \ldots, x_d)$ and n control points numbered from 1 to n. We suppose this program is written in a simple toy version of a C-like imperative language, comprising global variables, no procedures, assignments of variables using only *parallel assignments* $(x_1, \ldots, x_d) = T(x_1, \ldots, x_d)$, tests of the form $r(x_1, \ldots, x_d) \leq 0$, where $r : \mathbb{R}^d \mapsto \mathbb{R}$, and while loops with similar entry tests. We do not recapitulate the standard collecting semantics that associates to this program a monotone map $F : \left(\wp(\mathbb{R}^d)\right)^n \to \left(\wp(\mathbb{R}^d)\right)^n$ whose least fixed points $\mathrm{lfp}(F)$ has as ith component $(i = 1, \ldots, n)$ the subset of $\mathbb{R}^d$ of values that the d variables $x_1, \ldots, x_d$ can take at control point i. The aim of this section is to compute, inductively on the syntax, the abstraction (or a good over-approximation of it) $F^\sharp$ of F from $\mathbf{F}\left(\mathbb{P}, \overline{\mathbb{R}}\right)^n$ to itself defined as usual as:

$$F^\sharp(v) := (F(v^\star)^\dagger) \tag{3}$$

The notation $v^\star$ is in fact the vector of sets $(v_1^\star, \cdots, v_n^\star)$ and $(F(v^\star)^\dagger)$ is also interpreted componentwise.

We will detail Equation (3) for assignments and tests. We will remark that we will deal with constrained optimisation problem. We first present a classical method from optimisation theory to overapproximate our abstract semantics by a semantics with better properties. In second time, we will specify the method to assignments and tests.

3.1 Lagrange duality

Let f, $\{f_i\}_{i=1,\ldots,m}$ be functions on $\mathbb{R}^d$. Let us consider the following constrained maximization problem:

$$\sup\{f(x) \mid f_i(x) \leq 0, \forall i = 1, \ldots, m\} \tag{4}$$

In constrained optimization, it is classical to construct another constrained optimization problem from the initial one in order to solve an easier problem. A technique called Lagrange duality (for details see for example [AT03, Section 5.3]) consists in adding to the objective function the inner product of the vector of constraints with a positive vector of the euclidean space whose the dimension is the number of constraints. In our context, the value of (4) is given by the following sup-inf (primal) value (8):

$$\sup_{x \in \mathbb{R}^d} \inf_{\lambda \in \mathbb{R}^m_+} f(x) - \sum_{i=1}^m \lambda_i f_i(x) \ . \tag{5}$$

A simple result of constrained optimization called weak duality theorem ensures that if we commute the inf and the sup in the formula (8), the result is greater than the value (8). The commutation of the inf and the sup gives us the so called (dual) value:

$$\inf_{\lambda \in \mathbb{R}^m_+} \sup_{x \in \mathbb{R}^d} f(x) - \sum_{i=1}^m \lambda_i f_i(x) \ . \tag{6}$$

The vectors $\lambda \in \mathbb{R}^m_+$ are called vectors of Lagrange multipliers. The function $\lambda \mapsto \sup_{x \in \mathbb{R}^d} f(x) - \sum_{i=1}^m \lambda_i f_i(x)$ is always convex and lower semi-continuous, so it has good properties to minimize it. If the function f is concave, if the functions f_i are convex and if the Slater constraint qualification (i.e. there exists $x \in \mathbb{R}^d$ such that $f_i(x) < 0$ for all $i = 1, \ldots, m$) holds then (5) and (6) coincide.

3.2 Abstraction of assignments and test using Lagrange duality

3.2.1 Abstraction of assignments

We focus on assignments $(x_1, \ldots, x_d) = T(x_1, \ldots, x_d)$ at control point i. We denote by $\mathbb{A}$ the set of such control points. Equation (3) translates in that case to (given that $v_{\ell(i)}$ defines the abstract value at control point $\ell(i)$, i.e. the breakpoint of the assignment acts on) when $p \in \mathbb{P}$ is fixed :

$$\left(F_i^\sharp(v)\right)(p) = \sup\{p \circ T(x) \mid q(x) - v_{\ell(i)}(q) \leq 0, \forall q \in \mathbb{P}\} \tag{7}$$

Introducing the evaluation functions i.e. the family of functions $\{\mathbf{e}_x, x \in \mathbb{R}^d\}$ from $\mathbb{P}$ to $\mathbb{R}$ such that $\mathbf{e}_x(p) = p(x)$, we can reformulate Equation (7) as:

$$F_\ell^\sharp(v) = \sup_{\{x \in \mathbb{R}^d \mid (v_{\ell(i)} - \mathbf{e}_x)(q) \geq 0, \ \forall\, q \in \mathbb{P}\}} \mathbf{e}_{T(x)}$$

We get a self-map $F^\sharp$ on $\mathbf{F}\left(\mathbb{P},\overline{\mathbb{R}}\right)$ and then the latter equation will be use to write fixed point equations.

At Equation (7), we recognize the constrained optimization problem (4) and we use Lagrange duality as in the first step of Subsection 3.1. In our case, Lagrange multipliers are some non-negative functions λ from $\mathbb{P}$ to $\mathbb{R}$. We thus consider the function which we will call the *relaxed* function:

$$F_i^{\mathcal{R}}(v) := \inf_{\lambda\in\mathbf{F}\left(\mathbb{P},\mathbb{R}_+\right)} \sup_{x\in\mathbb{R}^d} \mathbf{e}_{T(x)} + \sum_{q\in\mathbb{P}} \lambda(q)\left(v_{\ell(i)}(q) - q(x)\right) \ .$$

When we fix a template $p \in \mathbb{P}$, we have:

$$\left(F_i^{\mathcal{R}}(v)\right)(p) = \inf_{\lambda\in\mathbf{F}\left(\mathbb{P},\mathbb{R}_+\right)} \sup_{x\in\mathbb{R}^d} p(T(x)) + \sum_{q\in\mathbb{P}} \lambda(q)\left(v_{\ell(i)}(q) - q(x)\right) \ . \tag{8}$$

3.2.2 Abstraction of simple test

Now, we focus on a simple test and we write j the control point of the test. We denote by $\mathbb{I}$ the set of such control points We recall that a test is written as $r(x_1,\ldots,x_d) \leq 0$ where $r : \mathbb{R}^d \to \mathbb{R}$. We assume that the operation for the "`then`" branch has the form $x = T_{then}(x)$ and the operation for the "`else`" branch has the form $x = T_{else}(x)$ where T_{then}, T_{else}. To enter into the "`then`" branch, the abstract values of control point $\ell(j)$ must satisfy the test condition of the "`then`" branch, so the abstraction of a test is $F_j(X) = T_{then}(X_{\ell(j)}) \cap \{x \in \mathbb{R}^d \mid r(x) \leq 0\}$ for the "`then`" branch. For the "`else`" branch, we have similarly $F_{j+1}(X) = T_{else}(X_{\ell(j)}) \cap \{x \in \mathbb{R}^d \mid r(x) > 0\}$ which is equivalent to $F_{j+1}(X) = T_{else}(X_{j-1}) \cap \{x \in \mathbb{R}^d \mid -r(x) < 0\}$. However, we cannot use Lagrange duality with strict inequalities and so we replace the set $\{x \in \mathbb{R}^d \mid -r(x) < 0\}$ by the set $\{x \in \mathbb{R}^d \mid -r(x) \leq 0\}$ which is larger so we compute at least a safe overapproximation. When the function r is concave and the set $\{x \in \mathbb{R}^d \mid -r(x) < 0\}$ is non-empty, the closure of the former set coincides with the latter set. For the "`else`" branch, the abstraction is, finally, $F_{j+1}(X) = T_{else}(X_{\ell(j)}) \cap \{x \in \mathbb{R}^d \mid -r(x) \leq 0\}$. As we deal with arbitrary functions r, it is sufficient to show here how to deal with the equations at control point j and we simply write T instead of T_{then}. Equation (3) translated in the case of tests leads to (given that $v_{\ell(j)}$ defines the abstract value at control point $\ell(j)$, i.e. the breakpoint of the assignment acts on) when $p \in \mathbb{P}$ is fixed:

$$\left(F_j^\sharp(v)\right)(p) = \left(T\left(v^\star_{\ell(j)} \cap \{x \in \mathbb{R}^d \mid r(x) \leq 0\}\right)\right)^\dagger(p)$$

then, by a simple calculus:

$$\left(F_j^\sharp(v)\right)(p) = \sup\{p\circ T(x) \mid q(x) \leq v_{\ell(j)}(q)\ \forall\, q \in \mathbb{P},\ r(x) \leq 0\}. \tag{9}$$

Using the evaluation functions, we can reformulate Equation (9) as:

$$F_j^\sharp(v) = \sup_{\{x\in\mathbb{R}^d \mid (v_{\ell(j)} - \mathbf{e}_x)(q)\geq 0,\ \forall\, q\in\mathbb{P},\ r(x)\leq 0\}} \mathbf{e}_{T(x)}$$

in order to write fixed point equations. Using again Lagrange duality, we get the following relaxed problem:

$$F_j^{\mathcal{R}}(v) := \inf_{\substack{\lambda\in\mathbf{F}\left(\mathbb{P},\mathbb{R}_+\right)\\ \mu\in\mathbb{R}_+}} \sup_{x\in\mathbb{R}^d} \mathbf{e}_{T(x)} + \sum_{q\in\mathbb{P}} \lambda(q)\left(v_{\ell(i)}(q) - q(x)\right) - \mu r(x) \ .$$

When a template $p \in \mathbb{P}$ is fixed, we get:

$$\left(F_j^{\mathcal{R}}(v)\right)(p) = \inf_{\substack{\lambda\in\mathbf{F}\left(\mathbb{P},\mathbb{R}_+\right)\\ \mu\in\mathbb{R}_+}} \sup_{x\in\mathbb{R}^d} p(T(x)) + \sum_{q\in\mathbb{P}} \lambda(q)\left(v_{\ell(i)}(q) - q(x)\right) - \mu r(x) \ . \tag{10}$$

3.3 Abstraction of loops

We do not know yet how to interpret the equation at control point i where we collect the values of the variables before the entry in the body of the loop, at control point $\ell_1(i)$, with the values of the variables at the end of the body of the loop, at control point $\ell_2(i)$: $F_i(X) = X_{\ell_1(i)} \cup X_{\ell_2(i)}$, since we know now how to deal with the interpretation of tests. By using Equation (3), for $v \in \mathbf{F}\left(\mathbb{P},\overline{\mathbb{R}}\right)^n$, $F_i^\sharp(v) = (v^\star_{\ell_1(i)} \sqcup v^\star_{\ell_2(i)})^\dagger$. As for zones, we notice that the union of two such $\mathbb{P}$-convex functions $v_{\ell_1(i)}$ and $v_{\ell_2(i)}$ is directly given by taking their maximum on each element of the basis of functions $\mathbb{P}$. Nevertheless, during the fixed point

iteration (as in Section 5) the functions $v_{\ell_1(i)}$ and $v_{\ell_2(i)}$ are not necessarily $\mathbb{P}$-convex. Moreover, if we take the abstract semantics $F_i^\sharp(v)$, we do not have an infimum of linear forms (or at least a maximum of linear forms) on the abstract values $v_{\ell_1(i)}$ and $v_{\ell_2(i)}$, a formulation that we need. Finally, we relaxed the abstract semantics $F_i^\sharp(v)$, using Proposition 2.1, by the supremum itself and:

$$F_i^{\mathcal{R}}(v) = \sup(v_{\ell_1(i)}, v_{\ell_2(i)}) \ . \tag{11}$$

By this reduction, the map $v \mapsto F_i^{\mathcal{R}}(v)$ is monotone on $\mathbf{F}\left(\mathbb{P}, \overline{\mathbb{R}}\right)^n$, we formulate this result at Proposition 4.1. From now on, we denote $\mathbb{U}$ the set of coordinates such that the concrete semantics is a meet operation.

4 Properties of the relaxed semantics

The introduction of relaxed aims to get better computational properties of the semantics. We describe in this section the properties of the relaxed semantics which justify the using of the new semantics. In order to reduce the size of the paper, the proofs are skipped.

First, we show at Theorem 4.1 that the computation of an invariant from relaxed semantics will provide a safe over-approximation of the invariant of the abstract semantics.

Theorem 4.1. *Let i be a coordinate in $\mathbb{A} \cup \mathbb{I} \cup \mathbb{U}$. For all $v \in \mathbf{F}\left(\mathbb{P}, \overline{\mathbb{R}}\right)^n$,*

$$F_i^\sharp(v) \leq F_i^{\mathcal{R}}(v)$$

Furthermore, we prove monotonicty of the semantics. This property will be crucial to show that Policy Iteration provides more and more precise overapproximation of an invariant until a fixed point is reached.

Proposition 4.1. *For $i \in \mathbb{A} \cup \mathbb{I} \cup \mathbb{U}$, the map $v \mapsto F_i^{\mathcal{R}}(v)$ is monotone on the set $\mathbf{F}\left(\mathbb{P}, \overline{\mathbb{R}}\right)^n$.*

Let v be in $\mathbf{F}\left(\mathbb{P}, \overline{\mathbb{R}}\right)^n$. We introduce auxiliary functions to make appear some hidden properties.

- For $i \in \mathbb{A}$, we now define, for $p \in \mathbb{P}$, for $\lambda \in \mathbf{F}\left(\mathbb{P}, \mathbb{R}_+\right)$, $F_i^\lambda(v)$ by:

$$\left(F_i^\lambda(v)\right)(p) := \sum_{q \in \mathbb{P}} \lambda(q) v_{\ell(i)}(q) + V_i^\lambda(p) \ . \tag{12}$$

$$\text{where } V_i^\lambda(p) := \sup_{x \in \mathbb{R}^d} p \circ T(x) - \sum_{q \in \mathbb{P}} \lambda(q) q(x)$$

- For $i \in \mathbb{I}$, we define, for $\lambda \in \mathbf{F}\left(\mathbb{P}, \mathbb{R}_+\right)$ and $\mu \in \mathbb{R}_+$, $F_i^{\lambda,\mu}(v)$ by:

$$\left(F_i^{\lambda,\mu}(v)\right)(p) := \sum_{q \in \mathbb{P}} \lambda(q) v_{\ell(i)}(q) + V_i^{\lambda,\mu}(p) \tag{13}$$

$$\text{where } V_i^{\lambda,\mu}(p) := \sup_{x \in \mathbb{R}^d} p \circ T(x) - \sum_{q \in \mathbb{P}} \lambda(q) q(x) - \mu r(x) \ .$$

The relaxed functional can now be readily rewritten as follows.

Lemma 4.1. *For $i \in \mathbb{A}$ and $j \in \mathbb{I}$:*

$$\left(F_i^{\mathcal{R}}(v)\right)(p) = \inf_{\lambda \in \mathbf{F}\left(\mathbb{P}, \mathbb{R}_+\right)} \left(F_i^\lambda(v)\right)(p), \qquad \left(F_j^{\mathcal{R}}(v)\right)(p) = \inf_{\substack{\lambda \in \mathbf{F}\left(\mathbb{P}, \mathbb{R}_+\right) \\ \mu \in \mathbb{R}_+}} \left(F_j^{\lambda,\mu}(v)\right)(p) \ .$$

We recall some mathematical tools to get convergence proofs. Some definitions have been earlier given in the text, we give formal definitions here. All topological aspects are understood in the sense of $\mathbb{R}^d$-standard norm topology. When it is necessary, the norm will be specified.

Definition 4.1 (Lower/upper semi-continuous functions)**.** A function $f : \mathbb{R}^d \mapsto \overline{\mathbb{R}}$ is said to be *lower semi-continuous* if for all $\alpha \in \mathbb{R}$, the set $\{x \in \mathbb{R}^d \mid f(x) \leq \alpha\}$ is topologically closed. A function $g : \mathbb{R}^d \mapsto \mathbb{R}$ is said to be *upper semi-continuous* if $-g$ is lower semi-continuous.

A continuous function and (pointwise) supremum of lower semi-continuous functions are lower semi-continuous. Note that f is lower semi-continuous function iff for all $x \in \mathbb{R}^d$ and for all sequence x_n which converges to x that $f(x) \leq \liminf_n f(x_n)$. Since we have $\sup_{i\in I} -f_i = -\inf_{i\in I} f_i$, we conclude that the infimum of upper semi-continuous functions is upper semi-continuous. We also get that an upper semi-continuous function satisfies $\limsup_n f(x_n) \leq f(x)$ for all $x \in \mathbb{R}^m$ and for all sequence x_n which converges to x.

When f is upper semi-continuous and order-preserving function and $(x_n)_{n\geq 0}$ is a decreasing converging sequence to $x \in \mathbb{R}^d$ then $f(\inf_n x_n) = f(x) = \inf_n f(x_n)$. For a function g lower semi-continuous and order-preserving, we get $g(\sup_n x_n) = g(x) = \sup_n g(x_n)$ for all increasing converging sequence $(x_n)_{n\geq 0}$ to x.

Definition 4.2 (Level boundiness)**.** A function f is said to level bounded if for all $\alpha \in \mathbb{R}$, the set $\{x \in \mathbb{R}^d \mid f(x) \leq \alpha\}$ is bounded.

When level boundiness is coupled with lower-semicontinuity, all sublevel sets are closed and bounded and thus compact sets in a standard $\mathbb{R}^d$-norm topology. Lower semi-continuity and level boundiness are sufficient to existence of optimal solutions for minimisation problem.

Definition 4.3 (Convex/concave functions)**.** A function $f : \mathbb{R}^m \mapsto \mathbb{R}$ is said to be *convex* if for all $\alpha \in (0, 1)$, for all $x, y \in \mathbb{R}^m$, $f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha)f(y)$. A function $g : \mathbb{R}^m \mapsto \mathbb{R}$ is said to be *concave* if $-g$ is convex.

Convexity is exactly what we need when we want to minimise functions. Indeed, a local minimiser is in that case a global minimiser. When differentiability also holds, then local minimality is reduced to compute the zero of derivatives. The same principle can be applied for concavity and maximisation.

Lemma 4.2. *Let $f : \mathbb{R}^d \mapsto \mathbb{R} \cup \{+\infty\}$ be a convex, proper (there exists $x \in \mathbb{R}^d$ such that $f(x) \in \mathbb{R}$), lower semi-continuous and level bounded function. Then $\inf_{x\in\mathbb{R}^d} f(x)$ is finite and there exists $\bar{x}$ such that:*

$$f(\bar{x}) = \inf_{x\in\mathbb{R}^d} f(x) \ ,$$

and the set of optimal solutions ($\bar{x}$ such that $f(\bar{x}) = \inf_{x\in\mathbb{R}^d} f(x)$) is a (non-empty) convex compact set.

Definition 4.4 (Slater's condition)**.** Let $f : \mathbb{R}^d \to \mathbb{R}$ and $g : \mathbb{R}^d \mapsto \mathbb{R}^m$. A constrained maximization $\sup\{f(x) \mid g(x) \leq 0,\ x \in \mathbb{R}^m\}$ satisfies Slater's condition iff there exists $x_0 \in \mathbb{R}^m$ such that $g(x_0) < 0$ i.e. for all coordinates $i = 1, \ldots, m$, $g_i(x_0) < 0$.

Slater's condition is linked to the non-emptyness of the interior of the set of constraints. Indeed if the interior of set of constraints is nonempty and constraints function g_i are convex and lower-semicontinuous, then $\text{int}(\{x \in \mathbb{R}^d \mid g(x) \leq 0\}) = \{x \in \mathbb{R}^d \mid g(x) < 0\}$, where int denotes the interior set and $g = (g_1, g_2, \ldots, g_m)$.

Depending on templates we choose, it is easy to check Slater's condition. For example, taking a set of templates $\mathbb{P}$ such that $p(0) = 0$ for all $p \in \mathbb{P}$, for $i \in \mathbb{A} \cup \mathbb{I}$, if $v_{\ell(i)}(p) > 0$, then Slater's condition holds for optimisation problems (7) and (9) (whenever $r(x_0) < 0$ is also satisfied for some x_0 such that $v_{\ell(i)}(p) > p(x_0)$).

Slater's condition is a sufficient condition to the existence of optimal solutions to the minimisation problem which appears in relaxed functional. Indeed Slater's condition implies the level boundiness of the dual functional. Optimal solutions will be used to compute a "pivoting" policy when a fixed point is not reached. Lemma 4.2 implies the following result.

Proposition 4.2 (Selection property)**.** *Let $i \in \mathbb{A}$. Assume that the maximisation problem* (7) *satisfies the Slater's condition and there exists $\lambda \in \mathbf{F}(\mathbb{P}, \mathbb{R}_+)$ such that:*

$$\sup_{x\in\mathbb{R}^d} \mathbf{e}_{T(x)} - \sum_{q\in\mathbb{P}} \lambda(q) q(x)$$

is finite. Then the minimisation problem (6) *admits a solution i.e. there exists $\lambda_p^* \in \mathbf{F}(\mathbb{P}, \mathbb{R}_+)$ such that:*

$$\big(F_i^{\mathcal{R}}(v)\big)(p) = p \circ T(x) + \sum_{q\in\mathbb{P}} \lambda_p^*(q)\big(v_{\ell(i)}(q) - q(x)\big)$$

Let $i \in \mathbb{I}$. Assume that the maximisation problem (9) *satisfies the Slater's condition and there exists $(\lambda, \mu) \in \mathbf{F}(\mathbb{P}, \mathbb{R}_+) \times \mathbb{R}_+$ such that:*

$$\sup_{x\in\mathbb{R}^d} \mathbf{e}_{T(x)} - \sum_{q\in\mathbb{P}} \lambda(q) q(x) - \mu r(x)$$

is finite. Then the minimisation problem (10) *admits a solution i.e. there exists* $(\lambda_p^*, \mu_p^*) \in \mathbf{F}(\mathbb{P}, \mathbb{R}_+) \times \mathbb{R}_+$ *such that:*

$$\big(F_i^{\mathcal{R}}(v)\big)(p) = p \circ T(x) + \sum_{q \in \mathbb{P}} \lambda_p^*(q)\big(v_{\ell(i)}(q) - q(x)\big) - \mu_p^* r(x)$$

The last result of this section discuss about continuity of the relaxed functional. Kleene iteration and policy iteration are iterative processes to compute fixed point. It is important to to prove that the limits of the sequences produced by both iterations scheme are fixed point. To show it, we need continuity.

Proposition 4.3 (Continuity result on $F_i^{\mathcal{R}}$)**.** *Let* $i \in \mathbb{A} \cup \mathbb{I} \cup \mathbb{U}$*. The following assertions holds:*

1. *Let* $p \in \mathbb{P}$*. The map from* $\mathbf{F}(\mathbb{P}, \mathbb{R})^n$ *to* $\overline{\mathbb{R}}$*,* $v \mapsto F_i^{\mathcal{R}}(v)(p)$ *is upper semi-continuous.*
2. *For all decreasing sequences* $(v_n)_{n \geq 0} \in \mathbf{F}\left(\mathbb{P}, \overline{\mathbb{R}}\right)^n$*:*

$$\left(\inf_{n \geq 0} F_i^{\mathcal{R}}(v_n)\right)(p) = \left(F_i^{\mathcal{R}}(\inf_{n \geq 0} v_n)\right)(p) \ .$$

3. *Let* $p \in \mathbb{P}$*. Let* $i \in \mathbb{A} \cup \mathbb{I}$*. Assume, there exists a nonempty compact set* $K_{i,p}$*:*
   - $i \in \mathbb{A}$, $\big(F_i^{\mathcal{R}}(\cdot)\big)(p) = \inf_{\lambda \in K_{i,p}} F_i^{\lambda}(\cdot)(p)$*;*
   - $i \in \mathbb{I}$, $\big(F_i^{\mathcal{R}}(\cdot)\big)(p) = \inf_{(\lambda,\mu) \in K_{i,p}} F_i^{\lambda,\mu}(\cdot)(p)$*;*

   Then:

   (a) the map from $\mathbf{F}(\mathbb{P}, \mathbb{R})^n$ *to* $\overline{\mathbb{R}}$*,* $v \mapsto F_i^{\mathcal{R}}(v)(p)$ *is lower semi-continuous,*

   (b) for all increasing sequences $(v_n)_{n \geq 0} \in \mathbf{F}\left(\mathbb{P}, \overline{\mathbb{R}}\right)^n$*:*

$$\big(\sup_{n \geq 0} F_i^{\mathcal{R}}(v_n)\big)(p) = \big(F_i^{\mathcal{R}}(\sup_{n \geq 0} v_n)\big)(p) \ .$$

5 Solving fixed point equations

5.1 Fixpoint equations in templates domain

We recall that $\mathbb{P}$ is a finite set of templates. The map F is a monotone map which interprets a program with d variables and n labels in $\wp\left(\mathbb{R}^d\right)^n$. We recall that $v^\star$ denotes the vector of sets $((v_1)^\star, \cdots, (v_n)^\star)$ and $F^\sharp(v) = (F(v^\star))^\dagger$ i.e. $\forall\, i,\ F_i^\sharp(v) = (F_i(v^\star))^\dagger$ and $F^{\mathcal{R}}$ is the map, the components of which are the relaxed functions of $F^\sharp$. As usual in abstract interpretation, we are interested in solving the least fixed point equation:

$$\inf\{v \in \mathrm{Vex}_{\mathbb{P}}(\mathbb{P} \mapsto \overline{\mathbb{R}})^n \mid F^\sharp(v) \leq v\} \tag{14}$$

Nevertheless, the function $F^\sharp$ is not easily computable (since the templates p are general). Hence, we solve instead the following fixed point equation in $\mathbf{F}\left(\mathbb{P}, \overline{\mathbb{R}}\right)^n$:

$$\inf\{v \in \mathbf{F}\left(\mathbb{P}, \overline{\mathbb{R}}\right)^n \mid F^{\mathcal{R}}(v) \leq v\} \tag{15}$$

We next describe and compare two ways of computing (or approximating) the smallest fixed point of the relaxed semantics equation: Kleene iteration in Section 5.2, and policy iteration in Section 5.3.

5.2 Kleene iteration

We denote by $\bot$ the smallest element of $\mathbf{F}\left(\mathbb{P}, \overline{\mathbb{R}}\right)^n$ i.e. for all $i = 1, \cdots, n$ and for all $p \in \mathbb{P}$, $\bot_i(p) = -\infty$. The Kleene iteration sequence in $\mathbf{F}\left(\mathbb{P}, \overline{\mathbb{R}}\right)^n$ is thus as follows:

1. $v^0 = \bot$
2. for $k \geq 0$, $v^{k+1} = F^{\mathcal{R}}(v^k)$

Now using continuity result of Proposition 4.3, we get the following theorem:

Theorem 5.1. *If for all* $i \in \mathbb{A} \cup \mathbb{I}$*, for all* $p \in \mathbb{P}$*, there exists a nonempty compact set* $K_{i,p}$*:*

- $i \in \mathbb{A}$, $\big(F_i^{\mathcal{R}}(\cdot)\big)(p) = \inf_{\lambda \in K_{i,p}} F_i^{\lambda}(\cdot)(p)$*;*
- $i \in \mathbb{I}$, $\big(F_i^{\mathcal{R}}(\cdot)\big)(p) = \inf_{(\lambda,\mu) \in K_{i,p}} F_i^{\lambda,\mu}(\cdot)(p)$*;*

then Kleene iteration converges to the smallest fixed point of $F^{\mathcal{R}}$*.*

Kleene iteration has the inconvenience that the values v^k which are obtained at a given iteration k (before convergence) do not provide a safe invariant. We shall see that policy iteration does not have this inconvenient: even if it is stopped at an intermediate step, it does provide a safe invariant. Moreover, the convergence of the Kleene iteration can be very slow, so it needs to be coupled with an acceleration technique which provides over-approximations. In our implementation, after a given number of iterations, and during a few iterations, we round bounds outwards with a decreasing precision (akin to the widening used in [GPBG08]).

5.3 Policy Iteration

We present now policy iteration algorithm. As usual, we present first the policies notion and then describe completely policy iteration at Algorithm 1.

5.3.1 Policy definition

A policy iteration algorithm can be used to solve a fixed point equation for a monotone function written as an infimum of a family of simpler monotone functions, obtained by selecting *policies*, see [CGG$^+$05, GGTZ07] for more background. The idea is to solve a sequence of fixed point problems involving simpler functions. In the present setting, we look for a representation of the relaxed function

$$F^{\mathcal{R}} = \inf_{\pi \in \Pi} F^{\pi} \tag{16}$$

where the infimum is taken over a set Π whose elements π are called *policies*, and where each function F^{π} is required to be monotone. The correctness of the algorithm relies on a selection property, meaning in the present setting that for each argument (i, v, p) of the function $F^{\mathcal{R}}$, there must exist a policy π such that $\big(F_i^{\mathcal{R}}(v)\big)(p) = \big(F_i^{\pi}(v)\big)(p)$. The idea of the algorithm is to start from a policy π, compute the smallest fixed point v of F^{π}, evaluate $F^{\mathcal{R}}$ at point v, and, if $v \neq F^{\mathcal{R}}(v)$, determine the new policy using the selection property at point v.

Let us now identify the policies. Lemma 4.1 shows that for each template p, each coordinate $F_i^{\mathcal{R}}$ corresponding to an assignment $i \in \mathbb{A}$ can be written as the infimum of a family of affine functions $v \mapsto F_i^{\lambda}(v)$, the infimum being taken over the set of Lagrange multipliers λ. The same lemma provides a representation of the same nature when the coordinate $i \in \mathbb{I}$ corresponds to a test, with now a couple of Lagrange multipliers (λ, μ). Choosing a policy π consists in selecting, for each $i \in \mathbb{A}$ (resp. $j \in \mathbb{I}$) and $p \in \mathbb{P}$, a Lagrange multiplier λ (resp. a pair of Lagrange multipliers λ, μ). We denote by $\pi_i(p)$ (resp. $\pi_j(p)$) the value of λ (resp. (λ, μ)) chosen by the policy π.

Then, the map F^{π} in (16) is obtained by replacing $F_i^{\mathcal{R}}$ by the affine functions appearing in Lemma 4.1, for $i \in \mathbb{A} \cup \mathbb{I}$. For coordinates corresponding to loops, i.e., $i \in \mathbb{U}$, we take $F_i^{\pi} = F_i^{\mathcal{R}}$ (the choice of policy is trivial) since the infimum operation does not appear in the expression of $F^{\mathcal{R}}$ (see Equation (11)).

Proposition 4.2 shows that the selection property is valid under a Slater constraint qualification condition. We thus introduce $\mathcal{FS}(P, \overline{\mathbb{R}})^n$, the set of elements of $\mathbf{F}\left(\mathbb{P}, \overline{\mathbb{R}}\right)$ which satisfy the Slater condition when the component F_i of F corresponds to an assignment or a test. More concretely: $v \in \mathcal{FS}(P, \overline{\mathbb{R}})^n$, if, for all $i \in \mathbb{A}$ the set:

$$\{x \in \mathbb{R}^d \mid q(x) < v_{\ell(i)}(q),\ \forall\, q \in \mathbb{P}\}$$

and, for $i \in \mathbb{I}$ and a test r, the set:

$$\{x \in \mathbb{R}^d \mid q(x) < v_{\ell(i)}(q),\ \forall\, q \in \mathbb{P}\} \cap \{x \in \mathbb{R}^d \mid r(x) < 0\}$$

are non-empty.

Note we can do restrictions on policies when degenerate cases appear:

- At some breakpoint i and for corresponding label j, if there exists $p \in \mathbb{P}$ such that $v_j(p) = -\infty$ then we can choose any vector of nonnegative λ such that $\lambda(p) \neq 0$. Note that in this case, $F_i^{\mathcal{R}}(v) \equiv -\infty$ and the smallest fixed point of $F^{\mathcal{R}}$ for the coordinate i must check $v_i \equiv -\infty$.
- At some breakpoint i and for corresponding label j, if there exists $p \in \mathbb{P}$ such that $v_j(p) = +\infty$ then we can choose any vector of nonnegative λ such that $\lambda(p) = 0$ for all $p \in \mathbb{P}$ such that $v_j(p) = +\infty$.

These two restrictions let us work with finite values when we have to compute optimal policies.

Algorithm 1 Policy Iteration in finite templates domain

1 Choose $\pi^0 \in \Pi$, $k = 0$.

2 Compute $V^{\pi^k} = \{V^{\pi^k}(q)\}_{q \in P}$ and define the associated function F^{π^k} by choosing λ and μ according to policy π^k using Equation (12) and Equation (13).

3 Compute the smallest fixpoint v^k in $\mathbf{F}\left(\mathbb{P}, \overline{\mathbb{R}}\right)^n$ of F^{π^k}.

4 If $w^k \in \mathcal{FS}(P, \overline{\mathbb{R}})^n$ continue otherwise return w^k.

5 Evaluate $F^{\mathcal{R}}(w^k)$, if $F^{\mathcal{R}}(w^k) = w^k$ return w^k otherwise take π^{k+1} s.t. $F^{\mathcal{R}}(w^k) = F^{\pi^{k+1}}(w^k)$. Increment k and go to 2.

5.4 Algorithm

For the third step of Algorithm 1, since $\mathbb{P}$ is finite and using Lemma 4.1, F^{π^l} is monotone and affine $\mathbf{F}\left(\mathbb{P}, \mathbb{R} \cup \{+\infty\}\right)^n$, we compute the smallest fixpoint of F^{π^l} by solving the following linear program see [GGTZ07, Section 4]:

$$\min \sum_{i=1}^{n} \sum_{q \in \mathbb{P}} v^i(q) \text{ s.t. } \left(F_k^{\pi_k^l}(v)\right)(q) \leq v_k(q),\ \forall k = 1, \cdots, n,\ \forall q \in \mathbb{P} \tag{17}$$

Remark 5.1. As in the case of the earlier policy iteration algorithms in static analysis [CGG+05, GGTZ07], an important issue is the choice of the initial policy, which may influence the quality of the invariant which is eventually determined. In [CGG+05, GGTZ07], the initial policy was selected by assuming that the infimum is the expression of the functional is attained by terms corresponding to guard conditions, see specially § 4.2 in [GGTZ07]. The same principle can be used here in presence of guards. At Section 6, we will detail the choice of initial policies when the set of templates is "good". Another method to choose an initial policy is to run a few Kleene iterations, in combination with an acceleration technique. This leads to a postfixpoint v of $F^{\mathcal{R}}$, and we select as the initial policy any policy attaining the infimum when evaluating $F^{\mathcal{R}}(v)$ (i.e., choose for $\pi_i(p)$ or $\pi_j(p)$ any Lagrange multiplier λ or pair of Lagrange multipliers λ, μ attaining the infimum in Lemma 4.1).

Remark 5.2. To ensure the feasibility of the solution of (17) computed by the LP solver, we replace, when possible, the constraint set by $F^{\pi^l}(v) + \epsilon \leq v$, where ϵ is a small constant (typically of the order of several $ulp(v)$, where $ulp(v)$, which stands for "unit of least precision", is the minimum over the coordinates i of the differences between the nearest floating points around v_i).

To obtain safe bounds even though we run our algorithm on machines which uses finite-precision arithmetic, we should use a guaranteed LP solver (e.g. LURUPA see [Kei05]) to check that the solution obtained verifies $F^{\pi^l}(v) \leq v$.

In [AGG10, AGG11], we have proved that policy iteration on quadratic templates converges towards a postfixpoint of our relaxed functional (Theorem 5.2 here). Combined with Theorem 4.1, this postfixpoint is also a postfixpoint of abstract semantics.

Theorem 5.2. *The following assertions hold:*

1. $F^{\mathcal{R}}(v^l) \neq v^l \implies F^{\mathcal{R}}(v^l) < v^l$;
2. *The sequence v^l computed by Algorithm 1 is strictly decreasing;*
3. *The limit v^∞ of the sequence v^l is a postfixpoint: $F^{\mathcal{R}}(v^\infty) \leq v^\infty$.*

Theorem 5.2 ensures that Algorithm 1 produces a sequence of safe overapproximations of the numerical invariant we want. Now we complete Theorem 5.2 by showing that actually, Algorithm 1 converges to a fixed point.

Theorem 5.3 (Convergence of policy iteration)**.** *If Slater condition is always satisfied then policy iteration converges to a fixed point.*

Proof. Third point of Theorem 5.2 is $F^{\mathcal{R}}(v^\infty) \leq v^\infty$. Now we have to prove that $v^\infty \leq F^{\mathcal{R}}(v^\infty)$.

At third step of Algorithm (1), we compute the smallest fixed point of F^{π^k}. Since we have for all $k \geq 0$, $v^{k+1} \leq v^k$ and by the fact that $F^{\pi^{k+1}}$ is order-preserving we have: $v^{k+1} = F^{\pi^{k+1}}(v^{k+1}) \leq F^{\pi^{k+1}}(v^k) =$

$F^{\mathcal{R}}(v^k)$. Now by taking the infimum on k, we get $v^\infty = \inf_k v^{k+1} = \inf_k v^k \leq \inf_k F^{\mathcal{R}}(v^k)$ and finally using the commutation of decreasing inf thanks to Proposition 4.3 then $\inf_k F^{\mathcal{R}}(v^k) = F^{\mathcal{R}}(\inf_k v^k) = F^{\mathcal{R}}(v^\infty)$ and we conclude that $v^\infty \leq F^{\mathcal{R}}(v^\infty)$. □

6 Templates design and initial policies

The choice of the initial policies is a crucial point for the quality of the fixed point found by Policy Iteration. For example, if we know that the values of the variables are bounded an unbounded first invariant can be a fixed point and Policy Iteration stops. The choice depends on the template design algorithm.

The set of reachable values taken by the variables of the analyzed program is bounded (in the sense of a $\mathbb{R}^d$-norm) iff there exists a function P such that P is level bounded ($\forall\, \alpha \in \mathbb{R}$, $\{x \in \mathbb{R}^d \mid P(x) \leq \alpha\}$ is bounded) and a sublevel of P is an invariant (i.e. contains all possible values taken by the variables of the analysed program). Nevertheless, finding both invariant function (relation) and invariant level seems to be difficult and in a first time, in template design, we only focus on invariant relations. It means that we are looking for a function such that *all* sublevels are invariant by program updates (assignments and guarded assignments). We can formulate the problem as follows.

Problem 6.1. *Find a function $P : \mathbb{R}^d \to \mathbb{R}$ such that:*

1. *For all $\alpha \in \mathbb{R}$ there exists $\beta \in \mathbb{R}_+$ such that $P(x) \leq \alpha \implies \|x\|_q^q \leq \beta$.*
2. *For all $i \in \mathbb{A}$, for all $v_i \in \mathbb{R}$, $P(x) \leq v_i \implies P(T_i(x)) \leq v_i$.*
3. *For all $i \in \mathbb{I}$, for all $v_i \in \mathbb{R}$, $r_i(x) \leq 0 \wedge P(x) \leq v_i \implies P(T_i(x)) \in v_i$;*

The norm $\|\cdot\|_q$ denotes the ℓ_q-norm on $\mathbb{R}^d$. In the first condition, we can take $\alpha = 1$ since the problem is homogeneous. Rewriting Problem 6.1 as constrained optimisation problem, the problem becomes to find a function $P : \mathbb{R}^d \mapsto \mathbb{R}$ such that:

1. there exists $\beta \in \mathbb{R}_+$, $-\infty < \sup\{\|x\|^2 \mid P(x) \leq 1\} \leq \beta$;
2. for all $i \in \mathbb{A}$, for all $v_i \in \mathbb{R}$, $-\infty < \sup\{P(T_i(x)) \mid P(x) \leq v_i\} \leq v_i$;
3. for all $i \in \mathbb{I}$, for all $v_i \in \mathbb{R}$, $-\infty < \sup\{P(T_i(x)) \mid P(x) \leq v_i,\ r_i(x) \leq 0\} \leq v_i$;

Using again Lagrange duality described at Subsection 3.1, a function $P : \mathbb{R}^d \mapsto \mathbb{R}$ would satisfy the relaxed problem (18), would be a solution of Problem 6.1.

$$\begin{cases} \exists\, \beta \in \mathbb{R}\ \text{ s.t }\ \inf_{\gamma \geq 0} \sup_{x \in \mathbb{R}^d} \{\|x\|^2 + \gamma(1 - P(x)) \leq \beta; \\ \forall\, i \in \mathbb{A},\ \forall\, v_i \in \mathbb{R}\ \text{ s.t }\ \inf_{\gamma^i \geq 0} \sup_{x \in \mathbb{R}^d} \{P(T_i(x)) + \gamma^i(v_i - P(x)) \leq v_i; \\ \forall\, i \in \mathbb{I},\ \forall\, v_i \in \mathbb{R}\ \text{ s.t }\ \inf_{\gamma_1^i, \gamma_2^i \geq 0} \sup_{x \in \mathbb{R}^d} P(T_i(x)) + \gamma_1^i(v_i - P(x)) - \gamma_2^i r(x) \leq v_i; \end{cases} \tag{18}$$

For the two last conditions, if we introduce existence and universal quantifiers, the inequalities must hold for all v_i. This implies that μ^i and μ_1^i must be equal to 1 in order to have for $i \in \mathbb{A}$ $(1-\mu^i)v_i = 0$ and for $i \in \mathbb{I}$, $(1-\mu_1^i)v_i = 0$. Indeed, otherwise, it suffices to take a small enough v_i to contradict the inequalities. Note that if Problem 18 has a solution then there exists a solution $\overline{P}$ which is greater than the q-power of the ℓ_q-norm. Indeed for a solution $(P^*, \beta^*, \gamma^*, \{\gamma_2^{i^*}\}_{i \in \mathbb{I}})$ of Problem 6.2, the function $\overline{P} = \gamma^* P + \overline{\beta} - \gamma^*$ for a positive real $\overline{\beta} \geq \beta^*$ such that $\overline{\beta} - \gamma^* \geq 0$ is also a solution of Problem 6.2. Finally, good templates are computed by solving Problem 6.2.

Problem 6.2. *Find $P : \mathbb{R}^d \to \mathbb{R}, \gamma_2^i \in \mathbb{R}_+$ such that:*

1. $\forall\, x \in \mathbb{R}^d,\ P(x) - \|x\|_q^q \geq 0$;
2. $\forall\, i \in \mathbb{A},\ \forall\, x \in \mathbb{R}^d,\ P(x) - P(T_i(x)) \geq 0$;
3. $\forall\, i \in \mathbb{I},\ \forall\, x \in \mathbb{R}^d,\ P(x) - P(T_i(x)) + \gamma_2^i r(x) \geq 0$;

Assume we want to analyse a program with exactly one while infinite loop and either conditional branchments or update inside. We claim that we can easily initialise Policy Iteration.

Claim 6.1 (Policy Iteration Initialisation). Let $(P, \beta, \gamma, \{\gamma_2^i\}_{i \in \mathbb{I}})$ a solution to Problem 6.2 such that P is greater than $x \mapsto \|x\|_q^q$, then the set of templates $\mathbb{P} = \{x \mapsto x_i^q, i = 1, \ldots, d\} \cup \{P\}$ can be intialised as follows.

- Let $i \in \mathbb{A}$ and $p' \in \mathbb{P}$.
$$\pi_i^0(p') = p \mapsto \begin{cases} 0 & \text{if } p \neq P \\ 1 & \text{if } p = P \end{cases}$$
- Let $i \in \mathbb{I}$ and $p' \in \mathbb{P}$.
$$\pi_i^0(p') = \left(p \mapsto \begin{cases} 0 & \text{if } p \neq P \\ 1 & \text{if } p = P \end{cases}, \gamma_2^i\right)$$

Moreover, the following polyhedron

$$\{v \in \mathbf{F}(\mathbb{P}, \mathbb{R})^n \mid F^{\pi^0}(v) \leq v\}$$

is nonempty and $\mathrm{lfp}(F^{\pi^0})$ has finite coordinates.

Recall that, for a linear discrete dynamical system $x := Ax$, a quadratic function $x \mapsto x^\intercal Lx$, where L is a symmetric matrix, is called Lyapunov function iff: L is positive definite i.e. $x^\intercal Lx > 0$ for all nonzero x and $L - A^\intercal LA$ is positive definite. Suppose there exists only one (convergent) linear update in the analyzed program without guards (condition on $i \in \mathbb{I}$ is removed), then $(x \mapsto x^\intercal Lx, 1, 1)$ is a solution of Problem 6 for every Lyapunov function $x \mapsto x^\intercal Lx$. An algorithm to compute automatically floating points certified Lyapunov functions for while infinite loops and one (guarded) affine update has been developed in [DM12].

7 Application to quadratic templates

7.1 Quadratic templates and Shor's semi-definite relaxation scheme

In this section, we instantiate the set $\mathbb{P}$ to only contain quadratic functions.

Definition 7.1. A template $p \in \mathbb{P}$ is a quadratic template iff it can be written as:

$$x \mapsto p(x) = x^T A_p x + b_p^T x,$$

where A_p is a $d \times d$ symmetric matrix (in particular A_p can be a zero matrix), x^T denotes the transpose of a vector x, b_p is a $\mathbb{R}^d$ vector.

Finding the maximal value of a non-concave quadratic function under convex or non-convex quadratic constraints is known to be an *NP-Hard* problem, see [Vav90] for a discussion of complexity issues in quadratic programming. Shor's relaxation scheme (see [TN01, Section 4.3.1] or Shor's original article [Sho87] for details) consists of over-approximating the value of a general quadratic optimization problem by the optimal value of a semi-definite programming (SDP for short) problem, the latter being computationally more tractable.

Indeed, SDP problems can be solved in polynomial time to an arbitrary prescribed precision by the ellipsoid method [GLS88]. More precisely, let $\epsilon > 0$ be a given rational, suppose that the input data of a semi-definite program are rational and suppose that an integer N is known, such that the feasible set lies inside the ball of the radius N around zero. Then an ϵ-optimal solution (i.e., a feasible solution the value of which is at most at a distance ϵ from the optimal value) can be found in a time that is polynomial in the number of bits of the input data and in $-\log \epsilon$. Moreover, an ϵ-solution of an SDP problem can also be found in polynomial time by interior point methods [NN94] if a strictly feasible solution is available. However, when the input is rational, no size on the bit lengths of the intermediate data is currently known, so that the term "polynomial time" for interior point methods is only understood in the model of computation over real numbers (rather than in bit model [GJ79]). The advantage of interior methods is that they are very efficient in practice. We refer the reader to [PR97] for more information.

Shor's relaxation scheme consists in computing the value (6) by solving a semi-definite program. We introduce the matrix $M(g)$, for a quadratic function g written as $x^T A_g x + b_g^T x + c_g$ and the matrix $N(y)$ for a real y defined as:

$$M(g) = \begin{pmatrix} c_g & \frac{1}{2} b_g^T \\ \frac{1}{2} b_g & A_g \end{pmatrix}, \text{ and } N_{1,1}(y) = y,\ N_{i,j}(y) = 0 \text{ if } (i,j) \neq (1,1) \tag{19}$$

Let $\preceq$ denote the Löwner ordering of symmetric matrices, so that $A \preceq B$ iff all eigenvalues of $B - A$ are non-negative.

When we fix $\lambda \in \mathbb{R}_+^m$, we have to maximize an unconstrained quadratic problem and the the maximum is finite iff there exists $\eta \in \mathbb{R}$ such that, for all $x \in \mathbb{R}^d$, $f(x) - \sum_{i=1}^m \lambda_i f_i(x) \leq \eta$ and since f, f_i are

quadratic functions, this is equivalent to $x^T(A_f - \sum_{i=1}^m \lambda_i A_{f_i})x + (b_f - \sum_{i=1}^m \lambda_i b_{f_i} + c_f - \sum_{i=1}^m \lambda_i c_{f_i} - \eta \leq 0$ for all $x \in \mathbb{R}^d$ which is equivalent to the fact that the matrix $M(f) + \eta N(-1) - \sum_{i=1}^m \lambda_i M(f_i)$ is negative semi-definite. Consequently, taking the infimum over $\lambda \in \mathbb{R}_+^m$, we recover the value (6).

In conclusion, Shor's relaxation scheme consists in solving the following SDP problem:

$$\underset{\substack{\lambda \in \mathbb{R}_+^m \\ \eta \in \mathbb{R}}}{\text{Min}}\ \eta \text{ s.t. } M(f) + \eta N(-1) - \sum_{i=1}^m \lambda_i M(f_i)] \preceq 0 \tag{20}$$

which is equal to the value (6), hence an over-approximation of the optimal value of the problem (4). We can use a verified SDP solver as VSDP [JCK07] to solve a SDP problem.

Another advantage to instantiate quadratic templates is that we remark that the functions of Lemma 4.1, V_i^λ and $V_i^{\lambda,\mu}$ are the value of an unconstrained quadratic maximization problem. So, the functions V_i^λ and $V_i^{\lambda,\mu}$ can be determined algebraically. Moreover, $V_i^\lambda(p)$ and $V_i^{\lambda,\mu}(p)$ can take the value $+\infty$ if the matrices associated to the quadratic functions $x \mapsto p \circ T(x) - \sum_{q \in P} \lambda(q) q(x)$ and $x \mapsto p \circ T(x) - \sum_{q \in P} \lambda(q) q(x) - \mu r(x)$ are not negative semi-definite. Furthermore, the latter matrices depend on $\lambda \in \mathbf{F}(\mathbb{P}, \mathbb{R}_+)$ and on a couple $(\lambda, \mu) \in \mathbf{F}(\mathbb{P}, \mathbb{R}_+) \times \mathbb{R}_+^d$. So, to ensure the finiteness of the value, it suffices to choose λ (or a couple (λ, μ) in the case of tests) such that the corresponding matrix is negative semi-definite. We denote by $A^\bullet$, the Moore-Penrose general inverse of A, which can be defined as $\lim_{\epsilon \to 0} A^T(AA^T + \epsilon Id)^{-1}$. The following proposition shows how to evaluate the functions V_i^λ and $V_i^{\lambda,\mu}$. We only consider the evaluation of $V_i^{\lambda,\mu}$ since the evaluation of the former function can be viewed as a special case of the evaluation of the latter.

Proposition 7.1. *Let i be in $\mathbb{I}$ and let an assignment T such that, for all $q \in \mathbb{P}$, $q \circ T$ is a quadratic function. Let p be in P and let (λ, μ) be a couple in $\mathbf{F}(\mathbb{P}, \mathbb{R}_+) \times \mathbb{R}_+$, we write:*

$$\begin{aligned}
\mathcal{A}_p(\lambda, \mu) &= A_{p \circ T} - \sum_{q \in P} \lambda(q) A_q - \mu A_r \\
\mathcal{B}_p(\lambda, \mu) &= b_{p \circ T} - \sum_{q \in P} \lambda(q) b_q - \mu b_r \\
\mathcal{C}_p(\lambda, \mu) &= c_{p \circ T} - \sum_{q \in P} \lambda(q) c_q - \mu c_r
\end{aligned}$$

If $(\lambda, \mu) \in \mathbf{F}(\mathbb{P}, \mathbb{R}_+) \times \mathbb{R}_+$ satisfies $\mathcal{A}(\lambda, \mu) \preceq 0$ and $\mathcal{B}_p(\lambda, \mu) \in \mathrm{Im}(\mathcal{A}(\lambda, \mu))$ then:

$$V_i^{\lambda,\mu}(p) = -\frac{1}{4} \mathcal{B}_p(\lambda, \mu)^T \mathcal{A}_p(\lambda, \mu)^\bullet \mathcal{B}_p(\lambda, \mu) + \mathcal{C}_p(\lambda, \mu).$$

Otherwise $V_i^{\lambda,\mu}(p) = +\infty$.

7.2 A detailed example

Now we give details on the harmonic oscillator of Example 1. The program of this example which is given at Figure 2 implements an Euler explicit scheme with a small step $h = 0.01$, that is, which simulates the linear system $(x, v)^T = T(x, v)^T$ with

$$T = \begin{pmatrix} 1 & h \\ -h & 1 - h \end{pmatrix}$$

We want to use the information of a Lyapunov function $\underline{L}$ of the linear system T to compute bounds on the values taken by the variables x and v of the simulation: the function $(x, v) :\mapsto (x, v) L (x, v)^T$ furnishes a Lyapunov function with

$$L = \begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix}$$

Recall that Lyapunov functions for linear updates are solution of Problem 6.2. We also use the quadratic functions $(x, v) \mapsto x^2$ and $(x, v) \mapsto v^2$ which corresponds to interval constraints. We introduce the set of templates $\mathbb{P} = \{\underline{x}, \underline{v}, \underline{L}\}$ and below the program it is described the semantic equations for all the three control points. The set of templates $\mathbb{P}$ is thus good set of templates in the sense of Section 6 and we can use Theorem 6.1. Now we are going to focus on the third coordinate of $\big(F^{\mathcal{R}}(v)\big)(p)$. Let us consider, for example, $p = \underline{x}$, we get: $\big(F_3^{\mathcal{R}}(v)\big)(\underline{x}) =$

$$\inf_{\lambda \in \mathbf{F}\left(\mathbb{P}, \mathbb{R}_+\right)} \sup_{(x,v) \in \mathbb{R}^2} \sum_{q \in P} \lambda(q) w_2(q) + (x, v) \left(\begin{pmatrix} 1 - \lambda(\underline{x}) & h/2 \\ h/2 & h^2 - \lambda(\underline{v}) \end{pmatrix} - \lambda(\underline{L}) L \right) (x, v)^T \tag{21}$$

```
x = [0,1];
v = [0,1]; [1]
h = 0.01;
while (true) { [2]
  u = v;
  v = v*(1-h)-h*x;
  x = x+h*u; [3] }
```

$$\begin{aligned} F_1^\sharp(w)(p) &= \{\underline{x}(x,v) \leq 1,\, \underline{v}(x,v) \leq 1,\, \underline{L}(x,v) \leq 7\} \\ F_2^\sharp(w)(p) &= (\sup\{w_1^\star(p), w_3^\star(p)\})^\dagger \\ F_3^\sharp(w)(p) &= \sup_{(x,v)\in(w_2)^\star} p(T(x,v)) \end{aligned}$$

Figure 2: Implementation of the harmonic oscillator and its semantics in $\mathbf{F}\left(\mathbb{P},\overline{\mathbb{R}}\right)^3$

By introducing the following symmetric matrices, we can rewrite (21) as Equation (22):

$$M(\underline{x}) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix},\ M(\underline{v}) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \text{ and } M(\underline{x}\circ T) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & h/2 \\ 0 & h/2 & h^2 \end{pmatrix}$$

$$\big(F_3^{\mathcal{R}}(w)\big)(\underline{x}) = \operatorname*{Min}_{\substack{\lambda\in\mathbf{F}\left(\mathbb{P},\mathbb{R}_+\right)\\ \eta\in\mathbb{R}}} \eta \text{ s.t. } M(\underline{x}\circ T) + \eta N(-1) + \sum_{q=\underline{x},\underline{v},\underline{L}} \lambda(q)\big(N(w_2(q)) - M(q)\big) \preceq 0 \qquad (22)$$

To initialize Algorithm 1, we choose a policy π^0 following Theorem 6.1. For the third coordinate of $F^{\mathcal{R}}$, we have to choose a policy π_3^0 such that $V_3^{\pi_3^0}(p)$ is finite for every $p = \underline{x}, \underline{v}, \underline{L}$. We can start, for example, by:

$$\pi_3^0(\underline{x}) = (0,0,1),\ \pi_3^0(\underline{v}) = (0,0,1),\ \pi_3^0(\underline{L}) = (0,0,1)\ .$$

This consists, for $p = \underline{x}$, in taking $\lambda(\underline{x}) = \lambda(\underline{v}) = 0$ and $\lambda(\underline{L}) = 1$ in (21). By Proposition 7.1 we find:

$$V_3^{\pi_3^0}(\underline{x}) = \sup_{(x,v)\in\mathbb{R}^2}(x,v)\begin{pmatrix} -1 & h/2-1 \\ h/2-1 & h^2-3 \end{pmatrix}(x,v)^T = 0$$

$$V_3^{\pi_3^0}(\underline{v}) = \sup_{(x,v)\in\mathbb{R}^2}(x,v)\begin{pmatrix} h^2-2 & h(1-h)-1 \\ h(1-h)-1 & (1-h)^2-3 \end{pmatrix}(x,v)^T = 0$$

$$V_3^{\pi_3^0}(\underline{L}) = \sup_{(x,v)\in\mathbb{R}^2}(x,v)(T^TLT - L)(x,v)^T = 0$$

The solution of the maximization problems are zero since all the three matrices are negative definite (i.e. a matrix B is negative definite iff $x^tAx < 0$ for all $x \neq 0$). The third matrix $T^TLT - L$ is negative definite since L satisfy the Lyapunov condition for the discrete linear system $(x,v) = T(x,v)$. To compute the least fixpoint of F^{π^0}, we solve the following linear program (see (17)):

$$\begin{gathered} \min \sum_{i=1}^{3}\sum_{p\in P}\beta_i(p) \\ \beta_2(\underline{L})\leq\beta_3(\underline{x}),\ \beta_2(\underline{L})\leq\beta_3(\underline{v}),\ \beta_2(\underline{L})\leq\beta_3(\underline{L}) \\ \beta_3(\underline{x})\leq\beta_2(\underline{x}),\ \beta_3(\underline{v})\leq\beta_2(\underline{v}),\ \beta_3(\underline{L})\leq\beta_2(\underline{L}) \\ 1\leq\beta_2(\underline{x}),\ 1\leq\beta_2(\underline{v}),\ 7\leq\beta_2(\underline{L}) \\ 1\leq\beta_1(\underline{x}),\ 1\leq\beta_1(\underline{v}),\ 7\leq\beta_1(\underline{L}) \end{gathered}$$

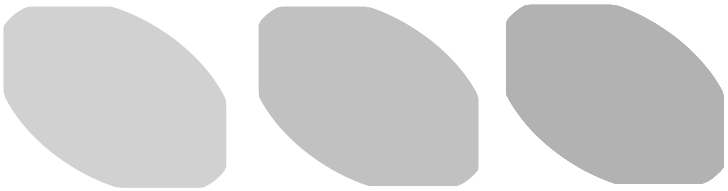
Using solver `Linprog`, we find:

$u_1^0(\underline{x}) =$	1.0000	$u_2^0(\underline{x}) =$	7.0000	$u_3^0(\underline{x}) =$	7.0000
$u_1^0(\underline{v}) =$	1.0000	$u_2^0(\underline{v}) =$	7.0000	$u_3^0(\underline{v}) =$	7.0000
$u_1^0(\underline{L}) =$	7.0000	$u_2^0(\underline{L}) =$	7.0000	$u_3^0(\underline{L}) =$	7.0000

$\{x^2 \leq 7,\ v^2 \leq 7,\ 2x^2 + 3v^2 + 2xv \leq 7\}$

$\{x^2 \leq 4.1723,\ v^2 \leq 2.7724,\ 2x^2 + 3v^2 + 2xv \leq 7\}$

$\{x^2 \leq 3.9386,\ v^2 \leq 2.5940,\ 2x^2 + 3v^2 + 2xv \leq 7\}$



$\{x^2 \leq 3.6079,\ v^2 \leq 2.3876,\ 2x^2 + 3v^2 + 2xv \leq 7\}$

$\{x^2 \leq 3.5051,\ v^2 \leq 2.3353,\ 2x^2 + 3v^2 + 2xv \leq 7\}$

$\{x^2 \leq 3.5000,\ v^2 \leq 2.3333,\ 2x^2 + 3v^2 + 2xv \leq 7\}$

Figure 3: Successive templates along policy iteration, at control point 2, for the harmonic oscillator.

Using again `Yalmip` with the solver `SeDuMi`, the vector w is not a fixpoint of $F^{\mathcal{R}}$, so we get the new following policy:

$$\pi_3^1(\underline{x}) = (0, 0, 0.596),\ \pi_3^1(\underline{v}) = (0, 0, 0.3961),\ \pi_3^1(\underline{L}) = (0, 0, 0.9946)\ .$$

Finally, after 5 iterations we find that the invariant of the loop i.e. $w_2^\star$ at control point 2 is the set:

$$\{x^2 \leq 3.5000,\ v^2 \leq 2.3333,\ 2x^2 + 3v^2 + 2xv \leq 7\}\ .$$

We draw $w_2^\star$ at each iteration of Algorithm 1 in Figure 3.

This method is to be compared with the classical Kleene iteration with widening. On this example, we find without widening $x^2 \leq 3.5000$, $v^2 \leq 2.3333$ and $2x^2 + 3v^2 + 2xv \leq 7$ in 1188 iterations whereas with the acceleration technique described Subsection 5.2 we find $x^2 \leq 6.0000$, $v^2 \leq 4.0000$ and $2x^2+3v^2+2xv \leq 10$ in 15 iterations.

8 Conclusion and Future Work

We define Policy Iteration algorithm in a general setting using a finite domain of templates. We prove that Policy Iteration algorithm converges to a fixed point of the relaxed semantics. This result allows us to use characterization tools as defined [AGG14] to check whether the solution found is the smallest solution of the relaxed functional. We should propose a syntactic analyse of a code to determine automatically a method to solve the relaxed functional and a corresponding policy when it is possible.

References

[AGG08] A. Adje, S. Gaubert, and E. Goubault. Computing the smallest fixed point of nonexpansive mappings arising in game theory and static analysis of programs. Technical report, arXiv:0806.1160, Proceedings of MTNS'08, Blacksburg, Virginia, July 2008.

[AGG10] Assalé Adjé, Stéphane Gaubert, and Eric Goubault. Coupling policy iteration with semidefinite relaxation to compute accurate numerical invariants in static analysis. In Andrew D. Gordon, editor, *ESOP*, volume 6012 of *Lecture Notes in Computer Science*, pages 23–42. Springer, 2010.

[AGG11] Assalé Adjé, Stéphane Gaubert, and Eric Goubault. Coupling policy iteration with semidefinite relaxation to compute accurate numerical invariants in static analysis. *Logical Methods in Computer Science*, 8(1), 2011.

[AGG14] Assalé Adjé, Stéphane Gaubert, and Eric Goubault. Computing the smallest fixed point of order-preserving nonexpansive mappings arising in positive stochastic games and static analysis of programs. *Journal of Mathematical Analysis and Applications*, 410(1):227 – 240, 2014.

[AT03] A. Auslender and M. Teboulle. *Asymptotic Cones and Functions in Optimization and Variational Inequalities*. Springer, 2003.

[BRCZ05] R. Bagnara, E. Rodríguez-Carbonell, and E. Zaffanella. Generation of basic semi-algebraic invariants using convex polyhedra. In C. Hankin, editor, *Static Analysis: Proceedings of the 12th International Symposium*, volume 3672 of *LNCS*, pages 19–34. Springer, 2005.

[CC77] P. Cousot and R. Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Conference Record of the Fourth Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 238–252, Los Angeles, California, 1977. ACM Press, New York, NY.

[CGG+05] A. Costan, S. Gaubert, E. Goubault, M. Martel, and S. Putot. A policy iteration algorithm for computing fixed points in static analysis of programs. In *Proceedings of the 17th International Conference on Computer Aided Verification (CAV'05)*, volume 3576 of *LNCS*, pages 462–475. Springer, 2005.

[Cou05] P. Cousot. Proving program invariance and termination by parametric abstraction, lagrangian relaxation and semidefinite programming. In *Sixth International Conference on Verification, Model Checking and Abstract Interpretation (VMCAI'05)*, volume 3385 of *LNCS*, pages 1–24. Springer, 2005.

[DM12] Thao Dang and Ian M. Mitchell, editors. *Hybrid Systems: Computation and Control (part of CPS Week 2012), HSCC'12, Beijing, China, April 17-19, 2012*. ACM, 2012.

[FA08] E. Feron and F. Alegre. Control software analysis, part II: Closed-loop analysis. Technical report, arXiv:0812.1986, 2008.

[Fer05] J. Feret. Numerical abstract domains for digital filters. In *International workshop on Numerical and Symbolic Abstract Domains (NSAD 2005)*, 2005.

[FF08] E Feron and Alegre F. Control software analysis, part I: Open-loop properties. Technical report, arXiv:0809.4812, 2008.

[GGTZ07] S. Gaubert, E. Goubault, A. Taly, and S. Zennou. Static analysis by policy iteration on relational domains. In *Proceedings of the Sixteenth European Symposium Of Programming (ESOP'07)*, volume 4421 of *LNCS*, pages 237–252. Springer, 2007.

[GJ79] M. R. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H.Freeman & Co Ltd, 1979.

[GLS88] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer, 1988.

[GPBG08] E. Goubault, S. Putot, P. Baufreton, and J. Gassino. Static analysis of the accuracy in control systems: Principles and experiments. In *Formal Methods for Industrial Critical System (FMICS 2007)*, volume 4916 of *LNCS*, pages 3–20, 2008.

[GS07a] T. Gawlitza and H. Seidl. Precise fixpoint computation through strategy iteration. In R. De Nicola, editor, *Programming Languages and Systems, 16th European Symposium on Programming, ESOP 2007*, volume 4421 of *LNCS*, pages 300–315. Springer, 2007.

[GS07b] T. Gawlitza and H. Seidl. Precise relational invariants through strategy iteration. In Jacques Duparc and Thomas A. Henzinger, editors, *Computer Science Logic, 21st International Workshop, CSL 2007, 16th Annual Conference of the EACSL, Lausanne, Switzerland, September 11-15, 2007, Proceedings*, volume 4646 of *LNCS*, pages 23–40. Springer, 2007.

[JCK07] C. Jansson, D. Chaykin, and C. Keil. Rigorous error bounds for the optimal value in semidefinite programming. *SIAM J. Numer. Anal.*, 46(1):180–200, 2007.

[Kei05] C. Keil. Lurupa - rigorous error bounds in linear programming. In *Algebraic and Numerical Algorithms and Computer-assisted Proofs*, 2005. `http://drops.dagstuhl.de/opus/volltexte/2006/445`.

[Mor70] J. J. Moreau. Inf-convultion, sous-additivé, convexité des fonctions numériques. *Journal Mathématiques de Pures et Appliquées*, 49:109–154, 1970.

[MOS04] M. Müller-Olm and H. Seidl. Computing polynomial program invariants. *Inf. Process. Lett.*, 91(5):233–244, 2004.

[NN94] Y. Nesterov and A. Nemirovski. *Interior point polynomial algorithms in convex programming*. Society for Industrial and Applied Mathematics, 1994.

[PR97] P.M. Pardalos and M.V. Ramana. Semidefinite programming. In *Interior Point Methods of Mathematical Programming*, pages 369–398. Kluwer Academic Publishers, 1997.

[RCK07] E. Rodríguez-Carbonell and D. Kapur. Automatic generation of polynomial invariants of bounded degree using abstract interpretation. *Sci. Comput. Program.*, 64(1):54–75, 2007.

[Roc96] R.T. Rockafellar. *Convex Analysis*. Princeton University Press, 1996.

[Rub00] A. M. Rubinov. *Abstract Convexity and Global optimization*. Kluwer Academic Publishers, 2000.

[SCSM06] S. Sankaranarayanan, M. Colon, H. Sipma, and Z. Manna. Efficient strongly relational polyhedral analysis. In E. Allen Emerson and Kedar S. Namjoshi, editors, *Verification, Model Checking, and Abstract Interpretation: 7^{th} International Conference, (VMCAI)*, volume 3855 of *LNCS*, pages 111–125, Charleston, SC, January 2006. Springer.

[Sho87] N. Shor. Quadratic optimization problems. *Soviet J. of Computer and Systems Science*, 25(6):1–11, 1987.

[Sin97] I. Singer. *Abstract Convex Analysis*. Wiley-Interscience Publication, 1997.

[SSM05] S. Sankaranarayanan, H. B. Sipma, and Z. Manna. Scalable analysis of linear systems using mathematical programming. In *Sixth International Conference on Verification, Model Checking and Abstract Interpretation (VMCAI'05)*, volume 3385 of *LNCS*, pages 25–41, January 2005.

[TN01] A. Ben Tal and A. Nemirowski. *Lecture on Modern Convex Optimization: Analysis, Algorithm and Engineering Applications*. SIAM, 2001.

[Vav90] Stephen A. Vavasis. Quadratic programming is in NP. *Information Processing Letters*, 36(2):73 – 77, 1990.